**ESTRUCTURA BREVE PARA LA REUNIÓN DE PRÁCTICAS ENNDE**

**1. Inicio (presentación breve y contexto del proyecto)**

*“Estamos desarrollando una web para ENNDE orientada a la visualización de modelos 3D con capas activables, pensada para el ámbito artístico y patrimonial. El proyecto nace desde cero, sin plantilla ni CMS, y está implementado con tecnologías web puras: HTML, CSS y JavaScript, con Three.js para la parte 3D.”*

**2. Decisiones clave tomadas**

* **Descartamos WordPress** (LocalWP + Elementor) por limitaciones para trabajo colaborativo y control técnico.
* Apostamos por un stack basado en **Vite + Node + GitHub + Three.js**, lo que nos da:
  + Velocidad y modularidad en desarrollo.
  + Control total sobre carga de modelos y renderizado.
  + Posibilidad de implementar lógica avanzada, como capas, HDRI, control de materiales, etc.

**3. Lo que ya está hecho**

* 🧱 Estructura base de carpetas y flujo de trabajo con index.html y viewer.html.
* 🖼️ Diseño visual inicial creado con **Figma**, replicado luego en código.
* 📦 Drag and Drop funcional que redirige al visor y carga modelos desde sessionStorage.
* 🌐 Implementación del visor 3D con:
  + Cámara, luces, controles y entorno HDRI.
  + Botón de restablecer materiales al estado original.
  + Menú lateral con controles visuales (color, roughness, metalness).
* 🧪 Soporte para **modelos GLB comprimidos** con Draco y pruebas de meshopt.

**4. Retos técnicos superados**

* 🧠 Entendimiento profundo del flujo Three.js: escena, cámara, materiales.
* 📂 Gestión de archivos 3D grandes (más de 30 MB) en local y su integración visual.
* 🤯 Control de conflictos con node\_modules en Git y uso efectivo de .gitignore.

**5. Próximos pasos**

* Añadir sistema de **capas activables** en el modelo.
* Mejorar interfaz de usuario del visor (más intuitiva y visual).
* Explorar opciones de exportación de escenas o capturas.
* Pulir detalles visuales del sitio y dejarlo listo para integración real con modelos de ENNDE.

**6. Actitud y compromiso**

*“Nos estamos implicando activamente, aprendiendo en el proceso, documentando cada paso y asegurando que el proyecto funcione de verdad. Queremos que esto no sea solo una práctica más, sino una web que pueda usarse y evolucionar.”*

**Explicación ligera de cada tecnología del segundo punto:**

**🔸 LOCALWP**

***Es una herramienta para crear sitios web en WordPress de forma local, pensada para usuarios menos técnicos.***👉 Nosotros lo probamos al principio, pero enseguida vimos que:

* Estábamos limitados a la estructura de WordPress.
* No podíamos trabajar con libertad el visor 3D.
* No ofrecía un control fino sobre cómo se cargaban o manejaban los modelos .glb.

**🔸 VITE**

***Es un entorno de desarrollo moderno para JavaScript y frameworks como React o Vue.*  
👉 En nuestro caso lo usamos porque:**

* Carga los cambios al instante sin recargar la página.
* Nos permite importar y usar módulos como Three.js y otros.
* Es muy ligero y rápido, ideal para pruebas visuales en tiempo real.

**🔸 NODE.JS**

***Es un entorno de ejecución para JavaScript en el servidor.*  
👉 Lo usamos para:**

* Gestionar dependencias (como Three.js o los cargadores Draco/meshopt).
* Tener un proyecto estructurado como cualquier proyecto moderno de desarrollo web.
* Usar herramientas como Vite o live-server con total compatibilidad.

**🔸 THREE.JS**

***Es la librería principal que usamos para cargar y mostrar los modelos 3D.*  
👉 Permite:**

* Cargar archivos .glb, .gltf, incluso comprimidos.
* Controlar la cámara, luces, materiales y animaciones.
* Crear escenas 3D con controles interactivos de forma muy visual.

**🔸 GITHUB**

***Es la plataforma donde subimos todo el proyecto para trabajar en equipo.*  
👉 Lo usamos para:**

* **Tener un historial de cambios.**
* **Evitar pisarnos al trabajar los dos a la vez.**
* **Compartir avances con facilidad, incluso desde diferentes ordenadores.**

**DESGRANANDO LA PRESENTACIÓN (ALGO MÁS EXTENSO)**

*“Estamos desarrollando desde cero una página web para la empresa ENNDE con el objetivo de visualizar modelos 3D. No usamos plantillas ni WordPress, sino que lo hacemos con HTML, CSS, JS y la librería Three.js para la parte gráfica 3D.”*

**Qué mostrar:**

✅ Abre index.html en el navegador.  
✅ Haz scroll general por el menú, botón de carga y diseño inicial.  
✅ Luego cambia a viewer.html brevemente para que vea que hay dos partes diferenciadas: carga y visualización.

**2. Decisiones clave tomadas**

**Qué decir:**

*“Probamos primero con LocalWP pero vimos que tenía muchas limitaciones. No podíamos trabajar cómodamente en equipo ni controlar cómo funcionaba internamente la carga de modelos. Por eso optamos por montar todo con Vite, Node, Three.js y GitHub para colaboración.”*

**Qué mostrar:**

✅ Enséñale brevemente tu estructura de carpetas del proyecto (donde se vea src, views, assets, node\_modules, etc.)  
✅ Muestra que usáis GitHub y menciona que trabajáis con ramas si surge.  
✅ Si hay tiempo: abre un package.json para que vea que es un proyecto con dependencias controladas.

**3. Lo que ya está hecho**

**Qué decir + mostrar en cada punto:**

* 🧱 **Estructura base**:  
  *“Tenemos el index.html para cargar el archivo y el viewer.html para mostrar el modelo 3D.”*  
  ✅ Enséñale la transición: sube un .glb y que vea cómo se abre el visor.
* 🖼️ **Diseño visual replicado de Figma**:  
  *“Hicimos un diseño base en Figma y lo trasladamos a código con una interfaz ligera pero clara.”*  
  ✅ Si tenéis el diseño en Figma o una captura, muéstrala. Si no, solo comentadlo.
* 📦 **Drag and Drop funcional**:  
  *“El usuario arrastra el archivo .glb y se guarda en sessionStorage para luego mostrarse en el visor.”*  
  ✅ Haz una prueba: arrastra un archivo en vivo.  
  ✅ Abre consola y muestra el sessionStorage.
* 🌐 **Visor 3D**:  
  *“El visor carga el modelo, permite mover la cámara, cambiar color, rugosidad, metalicidad... y restaurarlo.”*  
  ✅ Demuestra el cambio de color y sliders.  
  ✅ Pulsa el botón "Restablecer".
* 🧪 **Soporte para modelos comprimidos**:  
  *“Para poder trabajar con modelos grandes, añadimos soporte para GLB comprimidos con DRACO y meshopt.”*  
  ✅ Carga un modelo grande o uno que antes no funcionaba.

**4. Retos técnicos superados**

**Qué decir:**

*“Más allá de lo visual, hemos aprendido mucho sobre cómo funciona internamente Three.js, cómo manejar materiales, cómo solucionar conflictos en Git, o cómo cargar archivos grandes de forma eficiente sin reventar el navegador.”*

**Qué mostrar:**

✅ Abre una parte del código (scene.js) y señala funciones como loadModel(), restaurarMaterialesOriginales(), etc.  
✅ Si queréis, enseñadle un antes/después (modelo que no se cargaba y ahora sí).

**5. Próximos pasos**

**Qué decir:**

*“Nuestro siguiente objetivo es permitir que se activen o desactiven capas del modelo, para representar las diferentes partes como si fueran secciones o niveles. También queremos mejorar la interfaz, hacerla más visual e intuitiva.”*

**Qué mostrar:**

✅ Abre un modelo con varias mallas (si tenéis alguno) y señala visualmente cómo podría activarse/desactivarse.  
✅ Comenta que ya lo tenéis pensado a nivel técnico pero está pendiente de implementar.

**6. Actitud y compromiso**

**Qué decir:**

*“Estamos muy implicados, aprendiendo y aplicando en tiempo real. Todo está documentado en el diario y en las sesiones. No queremos que sea solo una práctica, sino una web funcional y útil para ENNDE si algún día se decide usarla.”*

**Qué mostrar:**

✅ Abre el .docx de sesiones o el diario.  
✅ Enseña que hay capturas, resumen de lo hecho, ideas anotadas.

**EXTRA: Opcional para “impactar”**

*“Incluso exploramos ya compatibilidad con modelos artísticos reales que podrían usarse, como este que bajamos de AR-Code.”*

✅ Muestra el modelo del cuadro en el visor funcionando.  
✅ Menciona que no todos los visores soportan esos modelos comprimidos, y que vosotros lo habéis conseguido.

**ENTONCES, ¿QUÉ ES VITE EXACTAMENTE?**

Vite **no es un IDE**, pero sí es un **entorno de desarrollo** en el sentido más técnico de la palabra. Es decir, es una **herramienta que prepara y gestiona el proyecto mientras desarrollas**, pero **no tiene interfaz gráfica ni editor propio**. Lo usarías junto con un editor como VS Code.

🔧 **Vite te da:**

* Un servidor local de desarrollo súper rápido.
* Soporte para módulos modernos de JavaScript (con import, por ejemplo).
* Recarga automática cuando haces cambios.
* Soporte para frameworks y librerías actuales (como React, Vue, Three.js…).

Piensa en él como:  
🧪 “El motor que hace que tu proyecto se comporte como si estuviera ya en producción, pero al momento y con recarga automática para que desarrolles más rápido”.